Data-Centric Storage in Sensornets

Scott Shenker
ICSI, Berkeley, CA 94704
shenker@icsi.berkeley.edu

Sylvia Ratnasamy*
ICSI/UCB, Berkeley, CA 94704
sylviar@cs.berkeley.edu

Brad Karp†
ICSI, Berkeley, CA 94704
bkarp@icsi.berkeley.edu

Ramesh Govindan
USC Comp. Sci., LA, CA 90089
ramesh@usc.edu

Deborah Estrin
UCLA Comp. Sci., LA, CA 90095
destrin@cs.ucla.edu

1. INTRODUCTION

Recent engineering advances in the design of small energy-efficient hardware and compact operating systems have enabled the development of large-scale distributed sensing networks (sensornets) made up of many small sensing devices equipped with memory, processors, and short-range wireless communication. These sensornets will provide an unprecedented amount of detailed measurements over wide geographic areas. However, these data are distributed across the entire sensornet, and so are hard to use. Communication between sensornet nodes requires the expenditure of energy, a scarce commodity in most sensornets. Thus, making effective use of sensornet data will require scalable, self-organizing, and energy-efficient data dissemination algorithms.

Since the content of the data is more important than the identity of the node that gathers them, sensornet researchers have found it useful to move away from the Internet's point-to-point communication abstraction and instead adopt abstractions that are more data-centric. This approach entails *naming* the data and using communication abstractions that refer to those names rather than to node network addresses [1, 9]. In particular, previous work on data-centric routing has shown it to be an energy-efficient data dissemination method for sensornets [10]. Herein, we propose a useful companion method, data-centric storage (DCS). In DCS, relevant data are stored by name at nodes within the sensornet; all data with the same general name (*e.g.,* elephant sightings) will be stored at the same sensornet node (not necessarily the one that originally gathered the data). Queries for data with a particular name can then be sent directly to the node storing those named data, without the flooding required in some data-centric routing proposals.

Several data-centric dissemination methods are conceivable, each with rather different performance characteristics. The appropriate data dissemination method for a particular task will depend on the nature of the sensornet, its intended deployment environment, and the expected workload. We make three principal contributions in this paper:

- We propose a novel data dissemination method, DCS, and show where it outperforms other approaches.
- We provide an organizing framework for comparing among three canonical data dissemination approaches, and predict where each performs best.
- We give an overview of GHT, a Geographic Hash Table system that implements DCS.

Our claim is not that data-centric storage is always the method of choice, but rather that under some conditions it will be the most desired option. In fact, we expect that future sensornets will embody all of these (or similar) data-centric dissemination methods, and that users will choose the appropriate method based on the task at hand. To understand the relative behavior of each dissemination method under different conditions, one must in turn understand the context in which these algorithms will be deployed.

For this reason, we begin our paper with an overview of related work (in Section 2) that gives a sense of the role played by data dissemination in a complete sensornet system. Thereafter, Section 3 provides a general discussion of sensornet dissemination algorithms—their constituent components and the environments in which they may be used. In Section 4 we describe three canonical dissemination methods and use a simple analytical model to compare their costs.

There may be many viable system designs for data-centric storage. We present a scalable DCS system design that builds on two recent advances: (1) the GPSR geographic routing algorithm [11] and (2) a new generation of efficient peer-to-peer lookup systems such as Pastry, CAN, Chord, and Tapestry [6, 18, 21, 24]. In Section 5, we describe the high-level design for our DCS system, which we call Geographic Hash Table (GHT). The design details and evaluation of GHT are described in a companion paper [19].

*Sylvia Ratnasamy's current affiliation: Intel Research, Berkeley, CA 94704.
†Brad Karp's current affiliation: Intel Research and Carnegie Mellon University, Pittsburgh, PA 15213, bkarp@cs.cmu.edu.

2. RELATED WORK

In this section, we briefly review related work on sensornets. We organize this discussion in "layers" ordered from bottom to top. These layers are used only to clarify the presentation and convey a sense of the role of data dissemination in a complete sensornet system; we don't mean to imply that sensornet architecture is organized into clean, well-separated layers. We begin our review at layer three (packet routing), as we are concerned with data dissemination, which interacts directly with layer three and above. Layers one (physical and OS) and two (low-level communication and self-configuration) are comparatively orthogonal to data dissemination.

L3: Packet routing:

Packet routing algorithms are required to deliver packets between nodes that are not in mutual radio range. Packet routing systems based on node identifiers are ill-suited to sensornets, where communication isn't addressed to node identifiers. It is expected that sensornets will instead implement geographic routing systems that deliver packets to nodes based on their location. Below we describe several types of geographic routing systems, each with its own communication abstraction and energy cost. In the following, we let n be the number of nodes in the sensornet, and assume that the diameter of the sensornet is approximately $O(\sqrt{n})$.

Strongly geographic routing algorithms, like GPSR [11], allow nodes to send to a particular location. To go from one random location to another requires $O(\sqrt{n})$ packet transmissions, which is our (approximate) metric for total energy consumption. *Weakly geographic* routing algorithms like GEAR [23] allow a node to send packets to a region and then distribute the packet within that region. The transmission costs here are $O(\sqrt{n})$ packet transmission to reach the region and $O(r)$ packet transmissions within the region, where r is the number of nodes in the region.

In addition to geographic routing, two other packet routing primitives are likely to be available in sensor networks. *Scoped flooding* algorithms flood to a limited region around the sending node. Their transmission cost is $O(r)$ where r is the number of nodes in the region. *Flooding* sends a packet to the entire sensornet, and requires $O(n)$ packet transmissions.

L4: Local collaborative information processing:

Event detection sometimes requires synthesizing results from several different sensors. The algorithms in this class only require collaboration between local nodes; *i.e.,* those that can be reached by a tightly-scoped flood. An example of such algorithms is described in [22].

L5: Wide-Area data dissemination:

Under the data-centric architecture for sensornets, data are named. The data dissemination methods we refer to here allow nodes and users to access data by name across the sensornet. Note that, in contrast, the local collaborative information processing only used data that could be found nearby; these wide-area data dissemination methods are needed to do collaborative processing in the wide area, as we describe below.

The most commonly mentioned wide-area data dissemination technique is *directed diffusion* [9, 10], an example of *data-centric routing*: routing decisions are based on the name of the data rather than on the identity of the sending and receiving nodes. We discuss directed diffusion at greater length in Section 4.3. In this paper we propose another data dissemination approach: *data-centric storage*, whereby event information is stored by name within the sensornet.

It should be noted that directed diffusion (and most other data-centric routing proposals) do not require any packet forwarding methods other than flooding. In contrast, the data-centric storage proposal we present here requires strongly geographic routing. Thus, the data dissemination method choice may be limited by the nature of the underlying packet routing mechanisms.

L6: Wide-area collaborative information processing:

These methods are akin to the local collaborative information processing methods mentioned above, except that the collaborating nodes need not be local. An example of such a collaboration is that required for tracking an object across a sensor field. In this case, scalable collaborative methods must be built on efficient wide-area data-dissemination algorithms. Zhao *et al.* [25] describe a collaborative tracking application built on top of directed diffusion.

L7: User-level tasking and querying:

The highest layer is where users insert their tasking and querying commands. An example of an approach that fits here is work that has been done on defining database semantics for queries on sensor networks [2, 8, 14].

3. ASSUMPTIONS AND TERMINOLOGY

This section lays out the context for the dissemination algorithms discussed in the following section. We first state our basic assumptions about the class of sensornets we consider and then describe some basic concepts regarding sensornet data and how they are organized.

3.1 Assumptions

Projected sensornet designs in the literature [5] differ greatly in their characteristics and intended uses. Here we focus attention on that class of sensornets where wide-area data dissemination is a needed service.

We consider large-scale sensornets with nodes that are spread out over some well-defined area, whose approximate geographic boundaries are known to network operators.

We assume nodes have short range communication, but are within radio range of several other nodes. We further assume that nodes know their own locations. GPS or some other approximate but less burdensome localization algorithm [3, 7, 16, 17, 20] provides this information. This assumption is crucial for our proposed data-centric storage mechanism. We believe it a reasonable assumption because in many cases the sensornet data are useful only if the locations of their sources are known.

We further assume that communication to the outside world takes place through one or more access points in the sensornet, and so getting data from a sensornet node to the outside world requires sending the data through the sensornet to the access point. We use the term *access path* to refer to the set of paths data take from sensor nodes to the access point(s). This assumption is not required by our DCS mechanism *per se*, but is part of our model for comparing costs of different dissemination mechanisms.

We assume energy is a scarce commodity for sensornet nodes [15], such that data dissemination algorithms should seek to minimize communication in order to extend overall system lifetime. While the mapping between communication and energy consumption is complicated—it depends greatly on the precise hardware involved and the packet transmission pattern—in what follows we will focus on two simplified metrics of energy consumption:

Total usage: The total number of packets sent in the sensornet

Hotspot usage: The maximal number of packets sent by any particular sensornet node

While in the rest of the paper we treat all nodes as having the same capabilities, it is likely that real sensornets will have a tiered architecture, where some nodes have very limited data storage capacity and others have much more significant storage (and perhaps also more battery power and better communication facilities). Our discussion applies to this tiered approach as well, as long as the these "macronodes" are numerous and widely dispersed [4].

These assumptions describe the physical environment of the sensornet. We next discuss how these sensornets might be used.

3.2 Observations and Events

The purpose of sensornets is to provide detailed sensing capabilities across a wide geographic area. The low-level readings from these sensors, which we call *observations*, are named (as described, for example, in [1, 9]). While sensornets give unprecedented access to detailed observations of the physical world, sending this overwhelming volume of observations directly to the access point(s) would quickly exhaust the energy reserves of the sensornet. Fortunately, in most cases users don't want the complete set of raw unprocessed data, but rather are more interested in specific *events*, such as earthquakes or animal sightings.

We use the term *events* to refer to certain pre-defined constellations of low-level observations. For example, detailed temperature and pressure readings might constitute observations, while a particular combination of temperature and pressure might define an "elephant-sighting" event. A sensornet system will be designed to detect several well-defined *types* of events. Typically, the large volume of observations prohibits communicating them directly to the outside world. Events are thus derived by processing the low-level observations within the sensornet through *collaborative information processing* techniques.

Events can be defined not only in terms of low-level observations but also in terms of other events. For instance, detecting an animal migration would involve many individual animal sightings. In general, there will be a *web* of events, with some events defined in terms of others. These events are not necessarily in a strict hierarchy, but in the context of a particular application there is some sense that some events are lower-level than others, and could be used to define the higher-level events.

3.3 Tasks, Actions, and Queries

The preceding discussion identified the various types of information—observations and events—that might be provided by sensornets. We now describe the operations used to manipulate these data.

Users send instructions (by flooding or some other global dissemination method) to sensornet nodes to run certain local identification *tasks*. These tasks could be simple, such as taking temperature readings, or complex, such as identifying an animal from a collection of sensor readings. In essence, one can think of tasks as downloaded code.

Once an event has been identified, nodes can take one of three *actions*: a node could send event information to external storage, store the event information locally, or use data-centric storage. Recall that data-centric storage involves storing the event information at a sensornet node that is chosen based on the event's name.[1] These three possible actions—external store, local store, and data-centric store—form the core of the three canonical approaches we describe in Section 4.

Unless the information has been sent to external storage, at this stage the event information is still not in the user's hands. *Queries* are used to elicit the event information from the sensornet. How queries are executed will depend on the actions nodes take upon event detection. If event information is stored locally then queries must be flooded to all nodes (unless the user has prior knowledge about the location of the event). If event information is stored using data-centric storage, the query can be sent to the sensornet node associated with that event name.

4. DATA-DISSEMINATION METHODS

The main goal in a data-dissemination algorithm is to extract relevant data efficiently from within the sensornet. In this section, we consider three *canonical* methods that combine the pieces described in the preceding section differently, yielding three very different approaches to sensornet design. We first describe these methods and then compare their costs analytically.

All the dissemination methods begin by flooding the tasks to the entire sensornet. The tasks are the set of identification instructions, specifying which events to detect, how to detect them, and what actions to take upon detection. We assume that the tasking instructions remain in force for long periods of time, so that the initial cost of issuing tasks is dominated by the cost of the ensuing data processing.[2]

We also assume that event locations are not known in advance and are distributed randomly throughout the sensornet. The case where this assumption does not hold is discussed in the following section.

Finally, in evaluating communication costs we assume asymptotic costs of $O(n)$ message transmissions for floods and $O(\sqrt{n})$ for point-to-point routing where n is the number of nodes.

4.1 Canonical Methods

Earlier we described three basic actions nodes could take upon detecting an event. These lead directly to three canonical sensornet methods.

External Storage (ES):

Upon detection of events, the relevant data are sent to external storage where they are further processed as needed. This entails a cost

[1] This approach, like all data-centric approaches, requires a naming scheme. We do not address this issue here, but merely note that the naming scheme is part of the definition of events and is supplied by the globally disseminated tasking instructions.

[2] In situations where tasks are short-lived, the cost of flooding tasks dominates all other costs, and it doesn't matter much which of the approaches below is used.

of $O(\sqrt{n})$ for each event. There is no cost for external queries since the event information is already external; queries generated by internal nodes in the process of event detection will incur a cost of $O(\sqrt{n})$ to reach the external storage.

Local Storage (LS):
Event information is stored locally (at the detecting node) upon detection of an event; this incurs no communication costs. Queries are flooded to all nodes at a cost of $O(n)$. Responses are sent back to the source of the query at a cost of $O(\sqrt{n})$ each.

Data-Centric Storage (DS):
Here, after an event is detected the data are stored by name within the sensornet. The communication cost to store the event is $O(\sqrt{n})$. Queries are directed to the node that stores events of that name, which returns a response, both at a cost of $O(\sqrt{n})$.

These three canonical methods have very different cost structures; we compare these analytically in the next subsection.

4.2 Approximate Communication Costs

This section uses a simple analytical model to derive approximate formulae for the communication costs for the three canonical methods; these formulae suggest which method is best suited for a particular sensornet workload.[3]

The cost structure for the canonical methods is described by several parameters. We consider a sensornet with n nodes equipped to detect T event types. We let D_{total} denote the total number of events detected, Q denote the number of event types for which queries are issued, and D_q denote the number of events detected for each event queried for. We assume there is no more than one query for each event type, so there are Q queries in total.

In comparing costs, we also consider the case where users only care about a summary of the events rather than a listing of each one; *e.g.,* one might just want a count of the number of elephants seen rather than a listing of each elephant sighting.

We compare costs using approximations for both the total number of packets and the packets arriving at the access point. The packet count at the access point is a good estimate of the hotspot usage, since we expect the access point to be the most heavily used area of the sensornet.

External Storage:
Total: $D_{total}\sqrt{n}$ Hotspot: D_{total}

Local Storage:
Total: $Qn + D_q\sqrt{n}$ Hotspot: $Q + D_q$

Data-Centric Storage:
Total: $Q\sqrt{n} + D_{total}\sqrt{n} + D_q\sqrt{n}$ (list)
Total: $Q\sqrt{n} + D_{total}\sqrt{n} + Q\sqrt{n}$ (summary)
Hotspot: $Q + D_q$ (list) or $2Q$ (summary)

In the above, (list) indicates a full listing of events is returned (requiring a packet for each event) and (summary) indicates only a summary of events is returned (requiring only one packet).

[3] In a companion paper [19], we verify the validity of these approximations through simulation.

These calculations suggest a few straightforward observations. First, if all other parameters are held fixed, then as n increases the local storage method incurs the greatest total packet count. Second, external storage always incurs a lesser total message count than data-centric storage, but the ratio $1 + \frac{Q+D_q}{D_{total}}$ is unlikely to be great if there are many events detected (and, if there is at least one event detected of each type, this ratio is bounded by 3). Third, if $D_q \gg Q$ and events are summarized, then data-centric storage has the least load (of all three methods) on the access path. Fourth, if events are listed and $D_{total} \gg D_q$ then data-centric storage and local storage have significantly lesser access loads than external storage.

These observations in turn suggest that data-centric storage is preferable in cases where (a) the sensornet is large, (b) there are many detected events and not all event types are queried, so that $D_{total} \gg \max[D_q, Q]$. This performance advantage increases further when summaries are used. However, if the number of events is large compared to the system size, $D_{total} > Q\sqrt{n}$, and event lists (rather than summaries) are used, then local storage may be preferable.

4.3 Additional Dissemination Methods

The three canonical methods described in the previous section certainly do not exhaust the design space; combinations of them yield hybrid methods specialized for particular needs. Examples of such combinations include:

Using Data-Centric Storage for Location Guidance:
For certain applications, one might combine LS and DCS by storing detailed event information locally and using DCS to inform a querier of an event's location so that subsequent queries can be directed to the proper local store.

Using Data-Centric Storage for Context:
In the course of processing local data, nodes may find it useful to have some context about global parameters. For instance, data-centric storage could give nodes access to the number of other animals sighted when a node is trying to determine if a migration is underway.

Geographically Targeted Queries:
The canonical methods are designed for cases where one doesn't *a priori* know the event location. If one already knows the location of the event through out-of-band techniques, then one can direct queries to that location using geographic routing methods (see [23]). This LS variant stores data locally, and queries are sent (at cost $O(\sqrt{n})$) to the relevant locations to retrieve the desired data. It avoids the cost of flooding in the canonical LS approach, and the cost of storing each event in the canonical DCS approach.

5. THE GEOGRAPHIC HASH TABLE

We now offer a brief overview of our design for the Geographic Hash Table (GHT) system that supports the data-centric storage abstraction; the detailed design and evaluation of GHT are described in [19].

GHT provides a *(key,value)*-based associative memory. Events are named with keys. Both the storage of an event and its retrieval are performed using these keys. GHT is naming-agnostic; any naming scheme that distinguishes the requisite events suffices. The operations GHT supports are:

Put(k, v) stores v (the observed data) according to the key k, the name of the data.

Get(*k*) retrieves whatever value is stored associated with key *k*.

In terms of functionality (*i.e.,* the above interface), GHT is inspired by the new generation of Internet-scale Distributed Hash Table (DHT) systems such as Pastry, CAN, Chord, and Tapestry [6, 18, 21, 24]. In these systems, nodes are assigned virtual identifiers and a data object's key is also hashed to a virtual identifier. Given a particular key, a name-based routing algorithm is then used to locate the node with virtual identifier closest to the key's virtual identifier. This node then serves as the storage node for that key.

Although GHT provides functionality equivalent to that of the DHTs, it would be inappropriate to adopt the DHT routing algorithms for use on sensornets. These algorithms typically interconnect nodes in a way determined by their logical identifiers in the DHT system, which are largely independent of their proximity in the physical network topology. On the Internet the IP routing system offers connectivity between nodes that are not topologically close. But in the energy-constrained sensornet environment, maintaining routing among all pairs of nodes is infeasibly expensive.

Instead, the core idea in GHT is to use the true geographic (*i.e.,* latitude-longitude) space occupied by the sensornet as the logical identifier space and use geographic routing as the underlying name-based routing technique. Thus, a key is hashed to a geographic position and geographic routing is used to locate the node physically closest to this position. This approach allows us to achieve the required hash-table-like functionality while working with only the true physical connectivity between nodes.

GHT uses GPSR [11], a geographic routing system for multi-hop wireless networks. Under GPSR, a sender that wishes to transmit packets to a destination node marks these packets with the destination's *position*.[4] All nodes know their own positions and the positions of nodes a single hop away from them. Using only this local knowledge, GPSR can route a packet to any destination node.

GHT, however, uses geographic routing in a slightly different manner. Under GHT, unlike under GPSR, the originator of a packet does *not* know the node that is the eventual destination of a packet. The originator of a GHT **Put()** or **Get()** for a key *k* hashes the name *k* into geographic coordinates that act as the destination of the packet for that operation. The hash function is ignorant of the placement of individual nodes in the topology; it merely spreads the different key names evenly across the geographic region where the network is deployed. Thus, it is quite likely that there is no node at the precise coordinates the hash function produces. Fortunately, the manner in which GPSR treats such a packet is precisely the behavior desired by GHT—GPSR forwards the packet until it reaches the node geographically closest to the destination coordinates. Under GHT, this closest node consumes and processes the packet. Note that GHT does not change the underlying GPSR routing algorithm; we merely leverage a previously unexploited characteristic of GPSR that allows all packets destined for an arbitrary location to be routed consistently to the node closest to it.

The above approach is sufficient to support our GHT interface provided sensornet nodes are completely stationary and reliable. However, as with DHTs, much of the subtlety in the GHT system design arises specifically to ensure robustness and scalability in the face of the many sorts of failures possible in a distributed system. GHT uses a novel *perimeter refresh protocol* to provide both persistence and consistency when nodes fail or move. This protocol replicates stored data for key *k* at nodes around the location to which *k* hashes, and ensures that the appropriate storage node for *k* is chosen consistently.

By hashing keys, GHT spreads storage and communication load between different keys evenly throughout the sensornet. When many events with the same key are stored, GHT avoids concentrating communication and storage at their shared home node by employing *structured replication*, whereby events with the same key can be divided among multiple mirrors.

GHT fundamentally requires that a node know its own geographic position. While this assumption seems reasonable for most sensornets, an open question is how (if at all) one might achieve DCS using only approximate geographic information, or better still, without requiring any position information at all. This question is the subject of our continuing research.

[4] A sender maps the destination's identifier to its current position using a location database, such as GLS [13].

6. CONCLUSION

We believe future sensornets will incorporate many different data dissemination mechanisms and users will choose among them based on the particular circumstances. In this paper we proposed data-centric storage as a novel approach to wide-area data dissemination. We identified the settings where data-centric storage may be the preferred method because of its energy efficiency. We provided a framework for reasoning about data dissemination that divides the design space into three canonical approaches, and gave a simple model for characterizing their respective energy costs.

We briefly described the design of GHT and our approach to achieving data-centric storage, but there are doubtless other approaches. In particular, the GRID location system (GLS) [13] can be extended to provide a similar capability. GLS constructs and maintains a distributed database that maps node addresses to geographic positions. While the goal in GLS is to provide a node location service, for our sensornet application, this database is merely an unnecessary level of indirection; we map event names directly to locations. The SCOUT [12] location tracking system might also be used in a similar manner. While SCOUT uses hierarchical addressing and routing based on landmark routing, GHT uses GPSR, a flat routing algorithm wherein nodes are addressed with geographic coordinates.

A Linux version of the GHT design for DCS is under development for iPAQs communicating with both 802.11 radios and mote radios. In our initial applications of the system, we will experiment with event naming schemes toward the goal of realizing the benefits of DCS fully.

7. REFERENCES

[1] W. Adjie-Winoto, E. Schwartz, and H. Balakrishnan, The Design and Implementation of an Intentional Naming System, In *Proceedings of the Symposium on Operating Systems Principles*, pp. 186–201, (Charleston, South Carolina, December 1999).

[2] P. Bonnet, J. Gehrke, and P. Seshadri, Querying the Physical World, IEEE Personal Communications Magazine, Special Issue on Networking the Physical World, (October 2000).

[3] N. Bulusu, J. Heidemann, and D. Estrin, GPS-Less Low Cost Outdoor Localization for Very Small Devices, IEEE Personal

Communications Magazine, Special Issue on Smart Spaces and Environments, (October 2000).

[4] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, Habitat monitoring: Application Driver for Wireless Communications Technology, In ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean, (Costa Rica, April 2001).

[5] Defense Advanced Research Projects Agency, Sensor Information Technology, `http://www.darpa.mil/ito/research/sensit`.

[6] A. Rowstron and P. Druschel, Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-peer Systems, In *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, (Heidelberg, Germany, November 2001).

[7] L. Girod and D. Estrin, Robust Range Estimation Using Acoustic and Multimodal Sensing, In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems*, (Maui, Hawaii, October 2001).

[8] R. Govindan, J. Hellerstein, W. Hong, S. Madden, M. Franklin and S. Shenker, The Sensor Network as a Database, preprint, (2002).

[9] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, Building Efficient Wireless Sensor Networks with Low-level Naming, In *Proceedings of the Symposium on Operating Systems Principles*, pp. 146–159, (Alberta, Canada, October 2001).

[10] C. Intanagonwiwat, R. Govindan, and D. Estrin, Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks, In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom)*, (Boston, Massachusetts, August 2000).

[11] B. Karp and H.T. Kung, Greedy Perimeter Stateless Routing, In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom)*, (Boston, Massachusetts, August 2000).

[12] S. Kumar, C. Alaettinoglu, and D. Estrin, SCalable Object-tracking through Unattended Techniques (SCOUT), In *Proceedings of the 8th International Conference on Network Protocols(ICNP)*, (Osaka, Japan, November 2000).

[13] J. Li, J. Jannotti, D. DeCouto, D. Karger, and R. Morris, A Scalable Location Service for Geographic Ad-hoc Routing, In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, (Boston, Massachusetts, August 2000).

[14] S. Madden, M. Shah, J. Hellerstein, and V. Raman, Continuously Adaptive Continuous Queries over Streams, In *Proceedings of the ACM SIGMOD Conference*, (Madison, WI, June 2002).

[15] G. Pottie and W. Kaiser, Wireless Integrated Sensor Networks (WINS): Principles and Approach, *Communications of the ACM*, Vol. 43, Number 5, pp. 51–58, (May 2000).

[16] N. Priyantha, A. Chakraborty, and H. Balakrishnan, The Cricket Location Support System, In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom)*, (Boston, Massachusetts, August 2000).

[17] N. Priyantha, A. Liu, H. Balakrishnan, and S. Teller, The Cricket Compass for Context-Aware Mobile Applications, In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom)*, (Rome, Italy, July 2001).

[18] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, A Scalable Content-Addressable Network, In *Proceedings of the ACM SIGCOMM*, (San Diego, California, August 2001).

[19] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, GHT: A Geographic Hash Table for Data-Centric Storage in SensorNets, In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, (Atlanta, Georgia, September 2002).

[20] A. Savvides, C.-C. Han, and M. B. Srivastava, Dynamic Fine-Grain Localization in Ad-Hoc Networks of Sensors, In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom)*, (Rome, Italy, July 2001).

[21] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications, In *Proceedings of the ACM SIGCOMM*, (San Diego, California, August 2001).

[22] K. Yao, R. Hudson, C. Reed, D. Chen, and F. Lorenzelli, Blind Beamforming on a Randomly Distributed Sensor Array, In *IEEE Journal on Selected Areas in Communication*, Vol. 16, Number 8, (October 1998).

[23] Y. Yu, D. Estrin, and R. Govindan, Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks, UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023, (May 2001).

[24] B. Y. Zhao, J. Kubiatowicz, and A. Joseph, Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing, Tech. Rep. UCB/CSD-01-1141, University of California at Berkeley, Computer Science Department, (April 2001).

[25] F. Zhao, J. Shin, and J. Reich, Information-Driven Dynamic Sensor Collaboration for Tracking Applications, In *IEEE Signal Processing Magazine*, (March 2002).